A Baseline Analysis for Podcast Abstractive Summarization

Chujie Zheng
chz@udel.edu
University of Delaware, USA

Harry Jiannan Wang
hjwang@udel.edu
University of Delaware, USA

Kunpeng Zhang
kpzhang@umd.edu
University of Maryland, USA

Ling Fan
lfan@tongji.edu.cn
Tongji University, China

ABSTRACT

Podcast summary, an important factor affecting end-users' listening decisions, has often been considered a critical feature in podcast recommendation systems, as well as many downstream applications. Existing abstractive summarization approaches are mainly built on fine-tuned models on professionally edited texts such as CNN and DailyMail news. Different from news, podcasts are often longer, more colloquial and conversational, and noisier with contents on commercials and sponsorship, which makes automatic podcast summarization extremely challenging. This paper presents a baseline analysis of podcast summarization using the Spotify Podcast Dataset provided by TREC 2020. It aims to help researchers understand current state-of-the-art pre-trained models and hence build a foundation for creating better models.

1 INTRODUCTION

The podcast industry has been dramatically growing and gaining massive market appeal. For example, Spotify spent approximately $200 million on the acquisition of Gimlet Media in 2019. However, the discovery and understanding of podcast content seem less progressive as compared to other types of media, such as music, movie, and news. This calls for more computationally effective methods for podcast analysis, including automatic summarization.

With the rapid development in Natural Language Processing, especially the success of attention mechanism and Transformer architecture [16], the text summarization task has received increasing attention and many models have been proposed to achieve good performance, especially in the news summarization field [8, 12, 17]. They are all trained and tested using well-known CNN and DailyMail (CNN/DM) dataset where the headlines are served as the ground truth of summaries.

In this short paper, the dataset we study is the recently released TREC 2020 Spotify Podcasts Dataset [3], which consists of 105,360 podcast episodes with audio files, transcripts (generated using Google ASR), episode summaries, and other show information. Different from news, podcasts have unique characteristics, such as lengthy, multi-modal, more colloquial and conversational, and nosier with contents on commercials and sponsorship, which makes podcast summarization task more challenging. In this study, we aim to share our preliminary results on data preprocessing and some baseline analysis, which is expected to empirically show the aforementioned data specialty and build a foundation for subsequent podcast analyses. The code and pre-trained models will be released after the TREC 2020 competition [1].

[1] https://github.com/chz816/podcast-summarization-baseline

2 DATA PREPROCESSING

The Spotify podcast dataset has 105,360 podcast episodes from 18,376 shows produced by 17,473 creators. The average duration of a single episode is 30 minutes, while the longest can be over 5 hours and the shortest is only 10 seconds. The TREC Podcast Track organizers form the "Brass Set" by cutting down the dataset to 66,245 podcast episodes using the following rules:

- Remove episodes with descriptions that are too long (¿ 750 characters) or too short (¡ 20 characters);
- Remove "duplicate" episodes with similar descriptions (by conducting similarity analysis);
- Remove episodes with descriptions that are similar to the corresponding show descriptions, which means the episode description may not reflect the episode content.

On top of the Brass Set, we impose several extra constraints to form a cleaner dataset as follows:

- Remove episodes with emoji-dominated descriptions, i.e., descriptions with less than 20 characters after removing emojis.
- Remove episodes longer than 60 minutes to control the length of the episode descriptions. This constraint can be easily altered or relaxed if necessary.
- Remove episodes with profanity language in the episode or show descriptions [12].
- Remove episodes with non-English descriptions.
- Remove episodes with sponsorship/advertisement-dominated descriptions.

After preprocessing, the dataset has 24,250 episodes left, which serves the dataset for all analyses in this study (see Table 1 for details).

3 BASELINE MODELS

The abstractive summarization task aims to automatically generate the podcast episode summaries based on the episode transcripts. The ground truth is the summary written by the podcast creators. The performance of summarization models is often measured using the ROUGE score [9], particularly the F1 scores of ROUGE-1, ROUGE-2, and ROUGE-L [2]. We also report recall (R) and precision (P).

We design two simple heuristic baselines for model comparisons:

- Baseline 1: Select the first k tokens from the transcript as the summary.

[2] https://pypi.org/project/pyrouge/

Dataset Preprocessing	# of Episodes
TREC Spotify Podcasts Dataset	105360
After filtering by the TREC organizer (Brass Set)	66245
After removing episodes with emoji-dominated descriptions	56977
After removing episodes longer than 60 minutes	48074
After removing episodes with profanity language	33329
After removing episodes with non-English descriptions	32993
After removing episodes with sponsorship/advertisement-dominated descriptions	24250

Table 1: Data Preprocessing and the Number of Episodes

- Baseline 2: Select the last k tokens from the transcript as the summary.

The idea behind both baselines is that the beginning or the end of the podcast may contain more important content information. Their performance is shown in Table 2, with k being varied between 100 and 500. We choose the maximum value of k to 500 because BERT [5] and other Transformer-based [16] models as we will discuss in the next section truncate the input to 512 tokens. The results exhibit an obvious pattern that longer summary tends to capture more words (measured by ROUGE-1) and phrases (measured by ROUGE-2 and ROUGE-L) that are also in the true summary, which often leads to higher recall but lower precision. The key takeaways are: (1). choosing k = 100 yields the best combined F1 score, which means 100 tokens (words) are long enough to capture the major summarization information. This is compatible with the distribution of the true summaries, where the average summary length is 44 and the maximal length is 144. (2). Baseline 1 has the highest F1 scores, which means the starting part of podcasts contains more useful and related information to podcast summaries than the ending part. This is also consistent with our observation that podcast episodes often give some overview at the beginning to tell the listeners what to expect.

4 SOTA MODEL EXPERIMENTS

In this section, we conduct a number of experiments for the podcast summarization task using three current state-of-the-art (SOTA) summarization models, including BART [8] [3], T5 [12], and ProphetNet [17]. More specifically, we use the pre-trained models, fine-tune them using the news datasets (CNN and DailyMail datasets [10]), and the preprocessed podcast dataset from Section 2. The goal is to get an overview idea about the performance of the SOTA models, which builds a foundation for better model innovation. All experiments are conducted under a machine with two Tesla V100 GPUs.

We split our processed podcast dataset into training, validation and testing sets by 60%, 20%, and 20% at random, resulting in 14,550 observations in the training set and 4850 observations in both validation and testing sets. Based on the baseline analysis in the previous section, we choose the beginning part of the episode transcripts as the input (we use the default settings that use 1024 tokens for BART and T5 and 512 tokens for ProphetNet) and the episode description from creators as the summarization ground truth. Table 3 shows the experiment results, from which we have the following observations.

(1). The performance of the SOTA models is comparable to the baseline models, which indicates that there is plenty of headroom for improvements and calls for more research in this emerging area.

(2). The F1 scores for ROUGE 1, 2, and L of ProphetNet on the CNN/DM dataset are 44.20, 21.17, 41.30, but the corresponding best F1 scores in Table 3 for the podcast dataset are only 26.76, 7.95, and 22.71. This huge performance gap implies that the podcast summarization task could be more challenging than the news headline summarization task due to the podcast's unique characteristics aforementioned.

(3). Fine-tuning the pre-trained models on the CNN/DM dataset for podcast summarization may result in lower performance compared with the vanilla pre-trained models, e.g. BART and ProphetNet. This urges us to think more about the lexicon differences between the podcast dataset and other existing datasets used in summarization tasks, such as CNN/DM, Gigaword [13], BigPatent [14], and PubMed[4].

We also provide some sample generated podcast summaries from different models in our repository.

Based on the baseline analysis in this paper, we discuss a number of directions for future research:

- Summarization based on long narrative structure: as discussed in [11], simple position heuristics are not sufficient for long narratives (such as podcast transcripts) summarization. How to define a narrative structure for better podcast summarization is interesting and worthy of the topic.
- Conversation summarization: podcasts are often conversational, colloquial, and multi-people. How to leverage existing research such as [6, 15, 18] to help podcast summarization is still largely missing.
- Multi-modal podcast analysis: the audio files of podcasts contain much richer information than the text transcripts, such as music, emotion, pitch, etc. We believe the multi-modal analysis is critical for podcast understanding and thus should play an important role in podcast summarization and recommendation [1].
- Long-document transformer: how to leverage recent research on [2], and [7] to potentially use the full podcast transcripts during training.

[3] In this paper, we use DistilBART provided by Hugging Face. It achieves better performance than the original BART model in our experiment.

Model		ROUGE-1			ROUGE-2			ROUGE-L		
		R	P	F	R	P	F	R	P	F
Baseline 1	k=100	38.07	**15.52**	**21.11**	8.18	**3.30**	**4.49**	33.20	**13.57**	**18.44**
	k=200	51.31	10.80	17.25	12.79	2.73	4.32	46.11	9.71	15.51
	k=300	58.01	8.28	14.09	15.59	2.28	3.83	53.01	7.57	12.87
	k=400	62.20	6.77	11.88	17.59	1.97	3.42	57.41	6.25	10.97
	k=500	**65.17**	5.76	10.31	**19.21**	1.76	3.10	**60.60**	5.36	9.59
Baseline 2	k=100	31.88	**13.08**	**17.76**	3.88	**1.62**	2.18	27.63	**11.39**	**15.43**
	k=200	44.25	9.32	14.89	6.58	1.42	**2.23**	39.43	8.32	13.27
	k=300	51.04	7.29	12.39	8.65	1.29	2.14	46.22	6.61	11.22
	k=400	55.45	6.04	10.59	10.42	1.20	2.04	50.78	5.53	9.70
	k=500	**58.68**	5.19	9.29	**11.93**	1.13	1.95	**54.18**	4.79	8.57

Table 2: Model Performance for Two Baseline Models

Model	ROUGE-1			ROUGE-2			ROUGE-L		
	R	P	F	R	P	F	R	P	F
Baseline 1 (k=100)	**38.07**	**15.52**	**21.11**	**8.18**	**3.30**	**4.49**	**33.20**	**13.57**	**18.44**
Baseline 2 (k=100)	31.88	13.08	17.76	3.88	1.62	2.18	27.63	11.39	15.43
DistilBART [8] [1]	30.02	19.44	22.26	6.26	4.20	4.73	26.05	16.98	19.39
DistilBART [8] + CNN/DM *	26.50	20.76	22.05	5.15	4.05	4.27	23.02	18.14	19.21
DistilBART [8] + Podcast **	32.36	**25.44**	**26.76**	9.28	**7.31**	7.67	27.36	**21.67**	**22.71**
T5 [12] [2]	25.74	19.39	20.59	4.75	3.52	3.75	22.14	16.80	17.77
T5 [12] + CNN/DM *	31.26	17.09	21.03	5.90	3.19	3.93	26.95	14.82	18.18
T5 [12] + Podcast **	31.66	18.43	22.15	6.46	3.72	4.46	24.91	14.59	17.49
ProphetNet [17] [3]	20.78	22.08	19.52	6.23	6.75	5.84	17.90	18.65	16.59
ProphetNet [17] + CNN/DM *	32.52	13.60	17.85	9.13	3.59	4.77	28.29	11.51	15.20
ProphetNet [17] + Podcast **	**34.26**	19.01	22.61	**12.37**	6.66	**7.95**	**29.57**	15.95	19.12

[1] DistilBART: we use Hugging Face Transformers (model: sshleifer/distilbart-cnn-12-6)
[2] T5: we use Hugging Face Transformers (model: t5-small)
[3] ProphetNet: we use released ProphetNet-large-160GB checkpoint
* Fine-tuned on CNN/DM Dataset
** Fine-tuned on Podcast Dataset

Table 3: Performance Comparison of Different Models

5 CONCLUSION

In this paper, we present the performance of podcast summarization using two baselines and SOTA models on the Spotify podcast dataset. We discuss several directions for future research in this field. We hope this pioneering baseline analysis and implementation can help researchers make more much-needed innovation in this exciting emerging research area.

REFERENCES

[1] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence* 41, 2 (2018), 423–443.

[2] Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150* (2020).

[3] Ann Clifton, Aasish Pappu, Sravana Reddy, Yongze Yu, Jussi Karlgren, Ben Carterette, and Rosie Jones. 2020. The Spotify Podcasts Dataset. *arXiv preprint arXiv:2004.04270* (2020).

[4] Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. *arXiv preprint arXiv:1804.05685* (2018).

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[6] Prakhar Ganesh and Saket Dingliwal. 2019. Abstractive summarization of spoken and written conversation. *arXiv preprint arXiv:1902.01615* (2019).

[7] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451* (2020).

[8] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).

[9] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.

[10] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023* (2016).

[11] Pinelopi Papalampidi, Frank Keller, Lea Frermann, and Mirella Lapata. 2020. Screenplay Summarization Using Latent Narrative Structure. *arXiv preprint arXiv:2004.12727* (2020).

[12] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683* (2019).

[13] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*

(2015).
[14] Eva Sharma, Chen Li, and Lu Wang. 2019. Bigpatent: A large-scale dataset for abstractive and coherent summarization. *arXiv preprint arXiv:1906.03741* (2019).
[15] Arpit Sood, Thanvir P Mohamed, and Vasudeva Varma. 2013. Topic-focused summarization of chat conversations. In *European Conference on Information Retrieval*. Springer, 800–803.
[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.
[17] Yu Yan, Weizhen Qi, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. *arXiv preprint arXiv:2001.04063* (2020).
[18] Xiaodan Zhu and Gerald Penn. 2006. Summarization of spontaneous conversations. In *Ninth International Conference on Spoken Language Processing*.